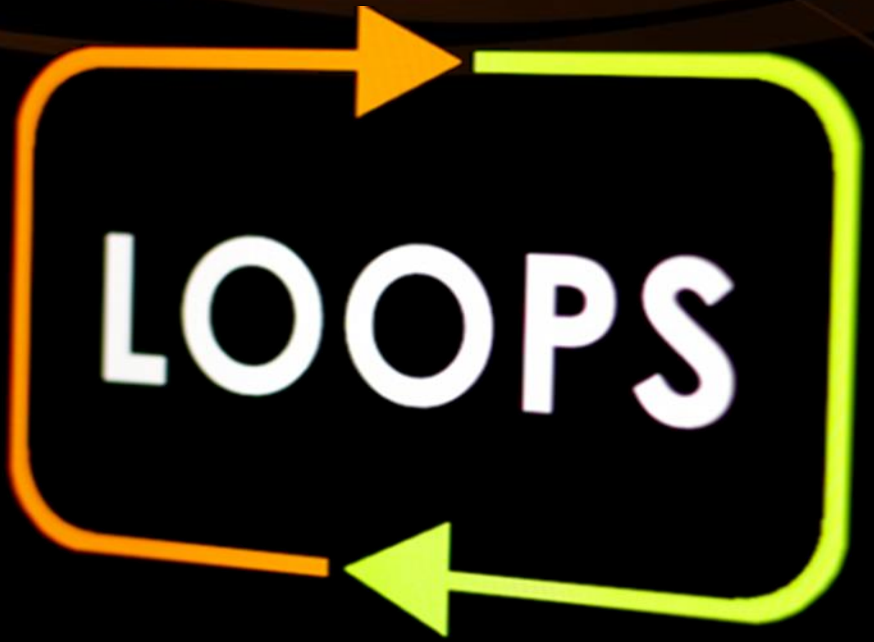
Повторения (цикли)

Прости повторения с for-цикъл





СофтУни
трейнърски екип
Софтуерен университет
http://softuni.bg

Have a Question?

SOFTWARE UNIVERSITY FOUNDATION

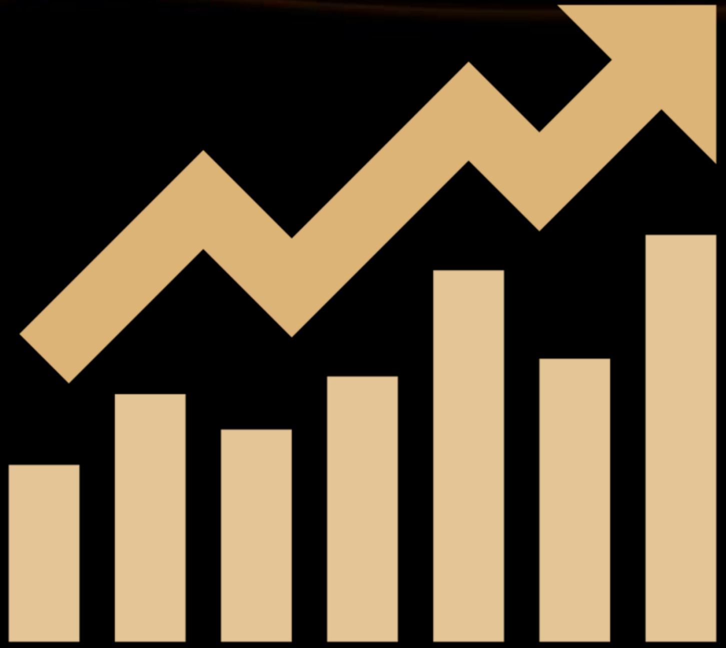
sli.do

12344321

Съдържание

1. Увеличаване и намаляване на стойността на променливи
2. For-цикъл
   - Конструкция





Увеличаване и намаляване на стойността на променливи

Увеличаване

- Инкрементиране – увеличаването на стойността на дадена променлива
  - Извършва се чрез оператори за инкрементиране – префиксни и постфиксни

Пример	Име	Резултат
++a	Пре-инкрементация	Увеличава стойността с единица и връща "**a**"
a++	Пост-инкрементация	Връща "**a**" и увеличава стойността с единица

  - Извършва се само върху променливи, които имат числена стойност

Увеличаване (2)

- Пре-инкрементация

```
int a = 1;
cout << ++a << endl;      // 2
cout << a << endl;        // 2
```

- Пост-инкрементация

```
int a = 1;
cout << a++ << endl;      // 1
cout << a << endl;        // 2
```

Намаляване

- Декрементиране – намаляването на стойността на дадена променлива
  - Извършва се чрез оператори за декрементиране – префиксни и постфиксни

Пример	Име	Резултат
--a	Пре-декрементация	Намалява стойността с единица и връща "**a**"
a--	Пост-декрементация	Връща "**a**" и намалява стойността с единица

  - Извършва се само върху променливи, които имат числена стойност

Намаляване (2)

- Пре-декрементация

```
int a = 1;
cout << --a << endl;        // 0
cout << a << endl;          // 0
```

- Пост-декрементация

```
int a = 1;
cout << a-- << endl;        // 1
cout << a << endl;          // 0
```



Повторения на блокове код

Конструкция за `for`-цикъл

Повторения (цикли) – for-цикъл

- В програмирането често пъти се налага да изпълним блок с команди **няколко** пъти
  - За целта използваме **цикли**

Ключова дума за конструкцията

Начална стойност

Крайна стойност

```
for (int i = 1; i <= 10; i++)
{
    cout << "i = " << i << endl;
}
```

Инкрементация на индекса (i)

Тяло: блок от код за изпълнение (за повторение)

Числа от 1 до 100

- Напишете програма, която:
  - Извежда числата [1, 100], всяко на нов ред
- Решение:

```
for (int i = 1; i <= 100; i++)
{
    cout << i << endl;
}
```

Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/532#0

Числа до 1000, завършващи на 7

- Напишете програма, която:
  - Извежда числата [1, 1000], които завършват на 7
- Решение:

```
for (int i = 0; i <= 1000; i++) {
   if (i % 10 == 7) {
      cout << i << endl;
}
```

Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/532#1

Числа до 1000, завършващи на 7 (2)


```
for (int i = 7; i <= 1000; i+=10)
{
    cout << i << " ";
}
```

ASCII таблица

- Символите, които използваме се представят като числа
  - Поместени са в ASCII таблицата
- Примери (знак и неговата ASCII стойност):

a	→	97	A	→	65
@	→	64	+	→	43

Пълна информация за ASCII

Всички латински букви - условие

- Напишете програма, която:
  - Извежда буквите от латинската азбука: [a, z]
- Решение:

```
for (char letter = 'a'; letter <= 'z'; letter++)
{
    cout << " " << letter << endl;
}
```

Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/532#2

Всички латински букви (2)

SOFTWARE UNIVERSITY FOUNDATION

- С помощта на ASCII Table можем да решим задачата по още един начин.
- Решение:

```
for (char i = 97; i <= 122; i++)
{
    cout << i << endl;
}
```

Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/532#2

Сумиране на числа - условие

SOFTWARE UNIVERSITY FOUNDATION

- Напишете програма, която:
  - Прочита цяло число(**n**) от потребителя
  - Прочита **n последователни** пъти числа и ги **сумира**
  - Извежда пресметнатата сума
  - Примерен вход и изход:

Вход		Изход
2 10 20	→	30
3 -10 -20 -30	→	-60
4 45 -20 7 11	→	43

Сумиране на числа - решение

```
int number;
cin >> number;
int sum = 0;

for (int i = 0; i < number; i++)
{
    int current_num;
    cin >> current_num;
    sum = sum + current_num;  // sum += num
}

cout << sum << endl;
```

Можем да четем данни в цикъл

Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/532#3

Най-голямо число - условие

- Напишете програма, която:
  - Прочита цяло число(**n**) от потребителя
  - Прочита **n последователни** пъти числа
  - Намира **най-голямото** измежду тях
- Примерен вход и изход:

```
2
100
99
```
→
```
100
```

```
3
-10
20
-30
```
→
```
20
```

```
4
45
-20
7
99
```
→
```
99
```

Най-голямо число - решение

```
int n, number; cin >> n;

int max = INT_MIN;

for (int i = 1; i <= n; i++){
     cin >> number;
     if (number > max){
          max = number;
     }
}
cout << max << endl;
```

#include <climits>

Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/532#4

Най-малко число - условие

- Напишете програма, която:
  - Прочита цяло число(**n**) от потребителя
  - Прочита **n последователни** пъти числа
  - Намира най-малкото измежду тях
- Примерен вход и изход:

```
2
100
99
```
→
```
99
```

```
3
-10
20
-30
```
→
```
-30
```

```
4
45
-20
7
99
```
→
```
-20
```

Най-малко число - решение

```
int n, number;
cin >> n;
int min = INT_MAX;

//TODO: Use logic similar to "Max number".
```

#include <climits>

Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/532#5

SOFTWARE UNIVERSITY FOUNDATION



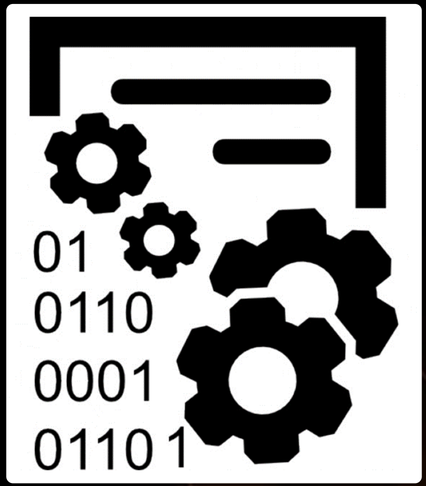

```
for (int i = 1; i < 100; i++)
{
    Console.WriteLine(i);
}
```

Повторения на блокове код

Работа на живо в клас (лаб)







Задачи с цикли

Техники за използване на for-цикли

Лява и дясна сума - условие

- Напишете програма, която:
  - Прочита цяло число(n) от потребителя
  - Прочита последователно **2*n** числа
  - Проверява дали сумите на левите **n** и десните **n** числа са равни
  - При равенство извежда "**Yes**"и сумата, в противен случай - "**No**" и разликата (изчислена като положително число)

Лява и дясна сума – условие (2)

- Примерен вход и изход:

```
2
10
90
60
40
```

```
Yes, sum = 100
```

```
2
90
9
50
50
```

```
No, diff = 1
```

Лява и дясна сума – решение

```
int number;
cin >> number;

int left_sum = 0;

for (int i = 0; i < number; i++)
    int current_num = 0;
    cin >> current_num;
    left_sum += current_num;

// TODO: read and calculate the right_sum
```

Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/532#6

```
if (left_sum == right_sum)
  cout << "Yes, sum = " << right_sum << endl;

else
  int diff = abs(left_sum - right_sum);
  cout << "No, diff = " << diff << endl;
```

Четна / нечетна сума - условие

- Напишете програма, която:
  - Прочита цяло число(n) от потребителя
  - Прочита последователно n на брой числа
  - Проверява дали сумата на числата на четни позиции е равна на сумата на числата на нечетни позиции
  - При равенство печата "**Yes**" + сумата; иначе печата "**No**" + разликата (положително число).

Четна / нечетна сума – условие (2)

- Примерен вход и изход:

```
4
10
50
60
20
```

```
Yes
Sum = 70
```

```
4
3
5
1
-2
```

```
No
Diff = 1
```

```
3
5
8
1
```

```
No
Diff = 2
```

Четна / нечетна сума – решение

```
int number; cin >> number;

int odd_sum = 0;
int even_sum = 0;
for (int i = 0; i < number; i++)
{
  int current_num; cin >> current_num;
  if (i % 2 == 0) even_sum += current_num;
  else odd_sum += current_num;
}

// TODO: print the sum / difference
```

Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/532#7

Сумиране на гласните букви - условие

- Да се напише програма, която въвежда текст (стринг)
  - Отпечатва сумата на гласните букви според таблицата по-долу:

a	e	i	o	u
1	2	3	4	5

  - Примери:

```
hello → 6
```
(e+o = 2+4 = 6)

```
hi → 3
```
(i = 3)

```
bamboo → 9
```
(a+o+o = 1+4+4 = 9)

```
beer → 4
```
(e+e = 2+2 = 4)

Сумиране на гласни букви – решение

```
string input;
getline(cin, input);
int sum = 0;
for (int i = 0; i < input.length(); i++)
    switch (input[i]) {
     case 'a': sum += 1; break;
     case 'e': sum += 2; break;
     // TODO: Add cases for other vowels.
     }
cout << sum << endl;
```

Можем да вземем **дължината** на текста

Можем да вземем **символ по индекс**

Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/532#8

Функцията putchar

- Използва се за принтиране
  - на буква

```
char letter = 'a';
putchar(letter);
cout << endl; //a
```

```
char letter = 'A';
putchar(letter);
cout << endl; //A
```

- на символ

```
char symbol = '$';
putchar(symbol);
cout << endl; //$
```

```
char symbol = '*';
putchar(symbol);
cout << endl; //*
```

- Когато дължината е повече от един символ или буква принтира последния срещнат.

```
char letters = 'ABCD';
putchar(letters);
cout << endl; //D
```

```
char symbols = '@#$%';
putchar(symbols);
cout << endl; //%
```

Принтиране на поредица от букви

```
char letters[] = "ABCD";

for (int i = 0; i < sizeof(letters) - 1; i++)
{
    putchar(letters[i]);
}

cout << endl; //ABCD
```

Преобразуване на малки и големи букви

SOFTWARE UNIVERSITY FOUNDATION

- От големи към малки – **tolower**

```
char letter = 'A';
putchar(tolower(letter));
cout << endl; //a
```

- От малки към големи - **toupper**

```
char letter = 'a';
putchar(toupper(letter));
cout << endl; //A
```

Еднакви думи – условие

- Напишете програма, която:
  - Прочита две думи от потребителя
  - Проверява дали те са еднакви(без да има разлика между главни и малки)
  - При равенство печата "**yes**"; иначе печата "**no**"

Еднакви думи – условие (2)

- Примерен вход и изход:

```
Hello
Hello
```
→
```
yes
```

```
Soft
Uni
```
→
```
no
```

```
SoftUni
softuni
```
→
```
yes
```

```
HeLlO
hELLo
```
→
```
no
```

Еднакви думи – решение

SOFTWARE UNIVERSITY FOUNDATION

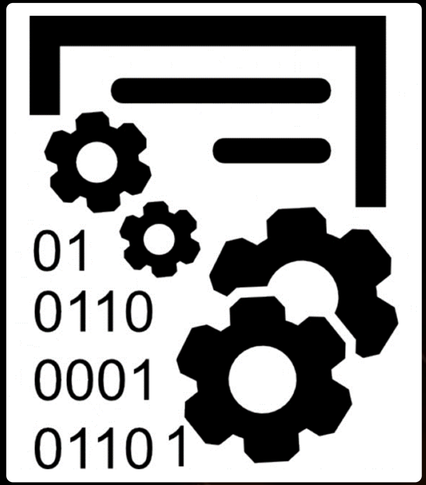
```
string word_one, word_two;
cin >> word_one >> word_two;

for (char i = 0; i < word_one.length(); ++i) {
    word_one[i] = tolower(word_one[i]);
    word_two[i] = tolower(word_two[i]);
}

if (word_one == word_two) cout << "yes" << endl;
else cout << "no" << endl;
```







Задачи с цикли

Работа на живо в клас (лаб)

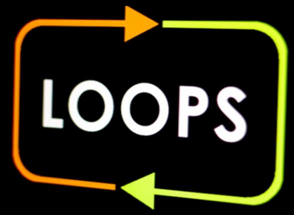
Какво научихме днес?

- Можем да инкрементираме/декрементираме числови стойности

```
int a = 1;
cout << --a << endl; // 0
cout << a++ << endl; // 0
cout << a << endl;   // 1
```

- Можем да повтаряме блок код с for-цикъл:

```
for (size_t = 1; i <= 10; i++)
{
    cout << "i = " << i << endl;
}
```

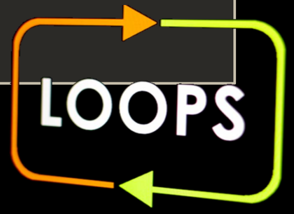




Какво научихме днес? (2)

- Можем да четем поредица от **n** числа от конзолата:

```
var n;
cin >> n;
for (int = 0; i < n; i++)
{ int num; cin >> num; …}
```



- Можем да вземем символ по индекс от текст

```
string text = "text";
char symbol = text[2];
cout << symbol << endl; // x
```



Какво научихме днес? (3)

- Можем да четем един символ или буква от конзолата:

```
char letter = 'b';
putchar(letter);
cout << endl; //b
```

```
char symbol = '^';
putchar(symbol);
cout << endl; //^
```

Повторения (цикли)



Въпроси?

https://softuni.bg/courses/programming-basics/

Лиценз

- Настоящият курс (слайдове, примери, видео, задачи и др.) се разпространяват под свободен лиценз "Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International"

- Благодарности: настоящият материал може да съдържа части от следните източници
  - Книга "Основи на програмирането със C#" от Светлин Наков и колектив с лиценз CC-BY-SA

Безплатни обучения в СофтУни

- Фондация "Софтуерен университет" – softuni.org
- Софтуерен университет – качествено образование, професия и работа за софтуерни инженери
  - softuni.bg
- СофтУни @ Facebook
  - facebook.com/SoftwareUniversity
- СофтУни форуми – forum.softuni.bg

